Final Report

Real Time Supervisors and Monitors for Performing Health Monitoring and Fault Detection for Systems Operating in Multiple Regimes
(STTR - Phase 1)

Contract Number: N00014-02-M0241

Submitted to:

Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5660

Submitted by:

Scientific Monitoring, Inc.
4801 S. Lakeshore Dr., Suite 103
Tempe, Arizona 85282-7156
(480) 752-7909, Fax: (480) 752-7866
www.scientificmonitoring.com

Collaborating Research Institution
Applied Research Laboratory, Pennsylvania State University

February 3rd, 2003

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT DATE	3. DATES COVERED *(From - To)*
03-02-2003	STTR Phase I Final Report	July 2002-January 2003

4. TITLE AND SUBTITLE
STTR Phase I Final Report: Real Time Supervisors and Monitors for Performing Health Monitoring and Fault Detection for Systems Operating in Multiple Regimes

5a. CONTRACT NUMBER
N00014-02-M0241

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Jaw, Link
Reichard, Karl
Kallappa, Pattada

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
001AD

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Scientific Monitoring Inc., 4801 South Lakeshore Drive, Tempe AZ 85281

Applied Research Lab, Pennsylvania State University, PO Box 30, State Colege PA 16804

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
Allen Moshfegh, ONR 351, Ballstone Tower One
800 North Quincy Street
Arlington VA 22217-5660

10. SPONSOR/MONITOR'S ACRONYM(S)
ONR

11. SPONSORING/MONITORING AGENCY REPORT NUMBER

12. DISTRIBUTION AVAILABILITY STATEMENT
Unlimited

13. SUPPLEMENTARY NOTES
None

14. ABSTRACT
In this Phase I STTR, SMI and ARL have developed a Real Time Supervisor for fault detection and system reconfiguration in a team of micro UAVs, that are tasked to perform a team mission like surveillance or rendezvous. We have developed a flexible, scalable and hierarchical architecture that will embed intelligence into the UAVs and enable them to detect functional level faults within them and enable team and mission reconfiguration to compensate for the fault. The architecture has been populated with intelligence and algorithms to enable these functions.

15. SUBJECT TERMS
Micro UAVs, Fault Detection, Real Time Supervisor, Intelligent Controls

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Pattada Kallappa
U	U	U	UU	22	19b. TELEPONE NUMBER (*Include area code*) 480-752-7909 x 110

NOTICE

THIS REPORT HAS BEEN REVIEWED AND APPROVED FOR PUBLICATION BY SCIENTIFIC MONITORING, INC. THE REPRODUCTION AND DISTRIBUTION OF THIS REPORT IS TO BE IN COMPLIANCE WITH THE CONTRACTED DIRECTIVE UNDER WHICH THIS REPORT HAS BEEN PRODUCED. REFERENCE SHOULD IN THE FIRST INSTANCE BE MADE TO THE CONTRACT, BUT CLARIFICATION CAN BE OBTAINED FROM THE REPORT'S AUTHORS.

AUTHOR: PATTADA KALLAPPA, Ph.D.
Scientific Monitoring, Inc
e-mail: kallappa@scientificmonitoring.com

AUTHOR: Karl Reichard, Ph.D.
PI: Applied Research Lab – Pennsylvania State University
e-mail: kmr5@psu.edu

REVIEWER
LINK JAW, Ph.D.
PI: Scientific Monitoring, Inc
e-mail: link@scientificmonitoring.com

1. Introduction

In Phase 1 the goal of the SMI and ARL-Penn State team was to develop a Real Time Supervisor that could be applied to a small autonomous UAV platform (drone), to provide it with the intelligence and capability to: detect faults and external threats; and reconfigure its behavior or mission to accommodate these faults and threats. It was also our goal to make this a scalable architecture that can be applied to a team of drones controlled from a non-vulnerable ground station (Tactical Coordinator/TC). A small Matlab/Simulink demonstration is being developed to provide proof of concept.

The drones on which this Real Time Supervisor will be applied are expected to function as a team and perform missions such as low altitude surveillance, search, or station keeping in hostile urban environments. The team of drones can function as a network and each drone can function as a node in the network. A survey of the available UAVs was conducted and the survey showed that Yamaha Rmax and BAI Aerosystems' Javelin are most suited for such applications. Other platforms being considered for this application include the class of micro-UAVs, which are ideally suited for urban reconnaissance missions and for operation as mobile communication nodes.

We have adopted and refined a behavior-based, intelligent control architecture to function as the Real Time Supervisor architecture for each drone. This intelligent control architecture will control and reconfigure missions at a functional level. It will communicate with lower level controllers for critical subsystems such as communications, flight control, navigation, or surveillance instrumentation. It will provide operating 'set points' to these lower level controllers. The architecture has two important modules: *perception* for analyzing sensor data to detect faults and threats; *response* for modifying or reconfiguring behavior, based on perception module outputs.

In order to demonstrate these concepts, we have simulated the application of the architecture to a Yamaha Rmax UAV. As a first step, we created a list of faults, external threats and configurable behaviors for this particular type of UAV. We developed algorithms to populate the perception module and enable it to detect faults and threats based on sensor data. The final goal of the simulation is to demonstrate the Real Time Supervisor architecture and algorithms for a team of two to three drones and a Tactical Coordinator. The simulation is a work in progress and will continue through Phase I option and into Phase II.

The following sections provide technical details of the work performed under this project. Section 2 provides a general description of the types of missions enabled by the technology developed under this project. The mission concept of operations was used to create the simulation testbed for testing the real time supervisor. Section 3 describes the Intelligent Control Architecture that forms the basis for the Real Time Supervisor. Section 4 provides a detailed description of the application of this architecture to a UAV. It includes a survey of small UAVs, with a tabulation of the hardware capabilities of these UAVs. It also contains a general description of the types of faults that may develop in UAV subsystems and a description of the actions, behaviors and capabilities of these UAVs. Section 5 provides a brief description of the Matlab™ Simulink™ simulation developed to test the fault detection and reconfiguration capabilities of this architecture.

2. Mission Concept of Operations

Military planners and tacticians anticipate future battles will likely take place in more urban settings with many obstacles and other hostile agents. The same types of scenarios must be accounted for in planning for emergency response in the event of natural or man-made disasters. Such operations will emphasize reduced reliance on humans and an increased use of robots and autonomous vehicles. An important ingredient in these combat and emergency response scenarios will be autonomous UAVs, because of their ability to avoid ground obstacles and provide an aerial snapshot of the battlefield for command, control, and communication. In the future, groups of autonomous vehicles will function as a team along the same lines as a platoon of soldiers working with each other, while maintaining communications to coordinate their respective missions roles and actions. Each team will have a specific mission in addition to individual mission roles. Each autonomous vehicle will require sufficient intelligence to react internal system faults and external threats. It will also need to communicate with other team members and a remote station that functions as the leader. This autonomous team may include any combination of autonomous, unmanned ground vehicles (UGV), air vehicles (UAV), surface ships (USV) or underwater vehicles (UUV).

The missions considered here consist of UAVs and UGVs with a remote, non-vulnerable Tactical Coordinator.

1. *Surveillance and transmission from hostile region* – The TC assigns a surveillance region to a group of drones. The TC organizes the drones and assigns them initial surveillance subregions. The drones are deployed and they survey their respective regions while collecting and transmitting data/pictures to TC. They stay in communication with each other and the TC. Once they have swept the entire region they return. This mission could be modified to involve recording pictures and terrain data and downloading it only after the entire region is surveyed.

2. *Search in hostile region* – The TC might need to search for a particular type of object, on the ground in hostile territory. The parameters of this task are similar to the surveillance task. The only difference being if the object is found the mission ends, otherwise it continues till the entire region is surveyed.

3. *Aerial attack* – At this juncture an autonomous mission of this type would involve targeting an entire area and blanketing it with ammunition. The mission parameter would be similar to mission 1. The drones will require some ammunition carrying capability.

In order to insure robust, theatre-wide communications, one possible role for the team of autonomous vehicles will be to function as a communications network, either stand-alone or as part of a larger network. Each autonomous vehicle will function as a node in the autonomous communications network. From a tactical viewpoint each UAV and ground vehicle is an intelligent agent performing offensive, defensive or surveillance functions.

In the envisioned missions, the autonomous drones will be required to communicate with each other, transmit data and move at low altitudes through crowded urban areas. They will be exposed to internal and external threats: damage, system faults or failures, and threat from hostile forces. The team and individual drones will need to adjust their behavior to compensate for these changes. Like all intelligent agents, each UAV must be able to adapt to changes in its environment and its own condition to optimize its functionality and maintain the ability to carry out its assigned role. This emphasizes the need for monitoring the UAV's external environment

and internal or self health; further, the autonomous system must be able to reconfigure its behavior based on environmental and health changes. However, in order to adjust to these changes in system and environmental status, the team and individual drones need to detect the faults and threats that cause these changes. An intelligent control system is required to detect, estimate and reconfigure operations.

The goal of this project is to create a real time supervisor that will provide the above mentioned health monitoring, environment sensing and behavior reconfiguring intelligence to this army of UAVs in the future urban battlefield. The UAVs along with the ground station form a two-layered hierarchical, distributed command and control structure of agents. Each UAV and the ground station function as a node. The hierarchical architecture is scalable and the level of hierarchies in this system can be increased to three or more. In this phase of the project, a two-layered hierarchy is used. Each node in this hierarchy has a real time supervisor that embeds intelligence into the node. Figure 1 shows a simple picture of a two level hierarchy with UAVs (drones with intelligence) and the ground station (leader/Tactical Coordinator).

.

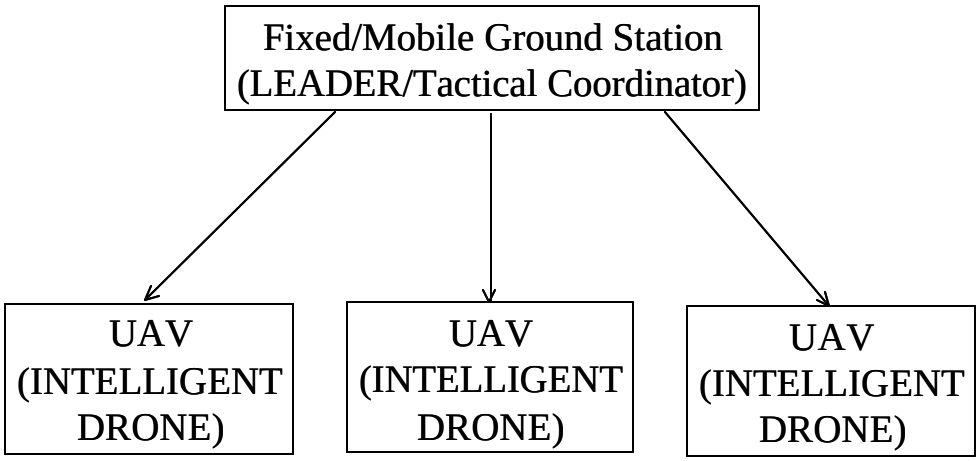

Figure 1: Schematic Diagram of Mission Hierarchy

The following assumptions will be made about the application platform(s):

1. The supervisor of the network of nodes is non vulnerable for the initial phase of the project. This implies that the network of drones will not be embedded with the capability to self organize or elect a leader. Once the real time supervisor is demonstrated successfully on this system the self organizing and supervisor election capabilities will be added to the group of unmanned vehicle nodes. This capability becomes critical when the supervisor is vulnerable.
2. Each drone has a vehicle control system that controls its position and velocity in a continuous time domain (the classical control problem) and sensors to perceive distances from obstacles.
3. Each drone functions as a communication node capable of rerouting itself in case of a breakdown with the supervisor. This concept has already been implemented at SRI in the AINS initiative.
4. The drones have a level of autonomy, where they receive directives from the supervisor and provide operational information to it, in return. The reactive control of the drones is not monitored by the supervisor.

In the future (Phase II) we plan on making the system reconfigurable so that any individual platform could assume the roll of TC – as long as it has the necessary resources. This will make the system more robust and tolerant to damage to any single unit.

3. Real Time Supervisor: Architecture

Many dynamical systems that need to be controlled are complex, large-scale, non-linear and operate in uncertain and unpredictable environments, and as a result are not amenable to accurate modeling. Moreover the conventional controllers are designed for the purpose of stabilization, regulation, tracking and performance optimization. The control needs of complex systems like teams of autonomous vehicles include requirements like safety, exception handling, multi system coordination, reconfigurability etc. So the conventional control techniques that are model based are not suitable for control and reconfiguration of systems. Intelligent controls offer alternatives in which control structure (architecture) and outputs in response to external commands, environmental conditions and internal status are determined by observed input/output behavior as opposed to mathematical or model based behavior of the plant. Typically this observed input behavior is modeled using binary logic, fuzzy logic, predicate calculus etc.

There is little to be gained by intelligent control when the plant model is well known and control requirements fall within the scope of conventional control requirements. For this reason the control is hierarchically structured (Figure 2), where at the lower level conventional control is exercised and at the higher level intelligent control is used. The tasks are delegated from higher level to lower level and sensory feedback is passed form lower level to higher level. The real time supervisor designed in this project will serve the needs of an intelligent controller.

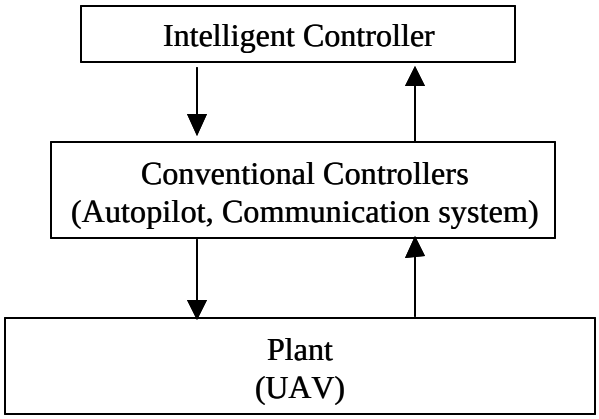

Figure 2: Control Hierarchy with a Real Time Supervisor

Some of the earlier architectures surveyed that have found their way into applications are the real-time control system (RCS) reference model architecture due to Albus and Meystel (1996). It consists of four subsystems, namely, sensor processing, value judgment, world modeling and behavior generation. This architecture provides a functional decomposition of intelligence. Brooks (1985) has designed the subsumption architecture. It is based on the idea of levels of increasing competence of an intelligent system, which need to be identified in the beginning of the design phase. This architecture provides the hierarchical concept to intelligent control architectures.

The real time supervisor architecture used here is behavior based and has a hierarchical character. In this architecture, the intelligent control function essentially uses sensor signals to calculate control actions. These control actions then serve as 'set points' for the lower level conventional controllers. This architecture is a cascade (Figure 3) of two main functional blocks, the perceptor and the response controller. On each side the functional blocks are book ended by input and output interfaces.

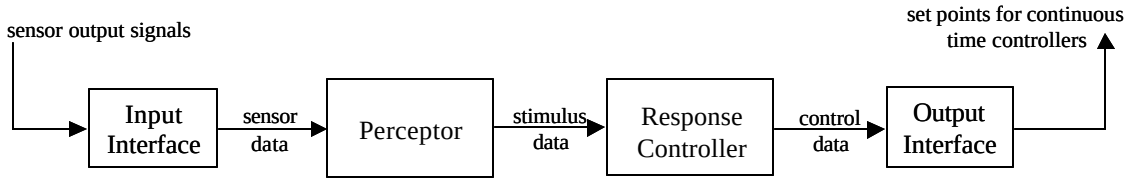

Figure 3: Cascade view of Real Time Supervisor

Input interface: The input interface collects sensor output. Its function is to convert the respective sensor output signal into sensor data. It collects this data into a data packet and feeds it to the perceptor. This relieves the perceptor module from having to deal with sensor interfaces.

Perceptor: The perceptor extracts relevant symbolic information from continuous sensor information to create an internal representation of the operating environment (threats), it's own internal state (fault and health status) and the mission status of the system. The perceptor module recommends actions or behaviors to the response module. This internal symbolic information space consists of a finite set of information states or instances. Some of these states are generic, while others are mission or platform specific. Examples include engine failure fault, battery power low or battery component fault, communication obstacle, communication complete, mission complete etc.

Operation of the perceptor module is depicted in Figure 4. The perceptor performs data fusion and classification to associate sensor information with new or existing information states. The fusion and classification portions of the perceptor module may employ rule bases, Boolean or fuzzy logic, neural and other forms of decision nets and data filters.

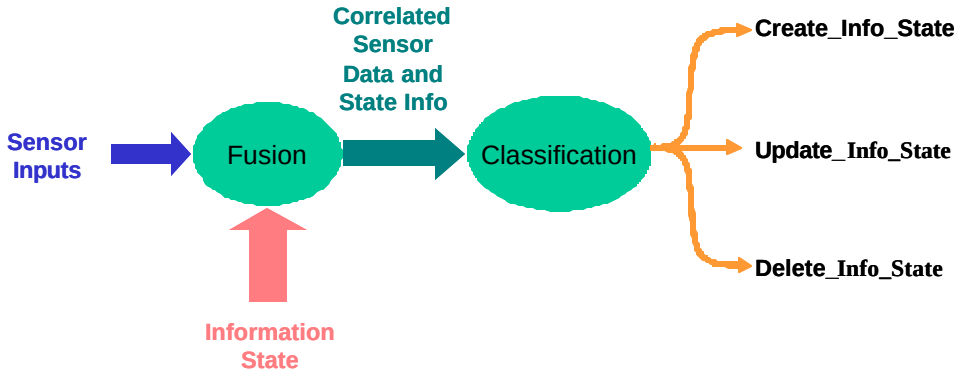

Figure 4: The perceptor module recommends actions and behaviors based in internal information states derived from sensor data.

The perceptor may use mathematical algorithms (Kalman filter, regression etc) for sensor fusion and it might use fuzzy and crisp logic, decision trees or decision nets to determine the stimuli. The Response module is a typical discrete event system, with internal states and state transitions, which trigger events.

Response: The response controller is a discrete event system that computes discrete control actions in response to discrete inputs/stimuli from perceptor and mission goals. The approach is to design a behavioral controller. Behaviors are certain high level activities that a system should exhibit in order to accomplish given mission goals. Behaviors can be configured in different modes and each behavior mode can be executed by the execution of activities called actions. Actions are primitives or atomic activities whose execution in a certain sequence results in the execution of a certain behavior mode. An action is a primitive in the sense that it specifies a unique set point for the lower level continuous time controllers that are controlled by the intelligent controller (Figure 3). A behavior could consist of one or more actions. One action could be a part of more than one behavior. The set of behaviors exhibited by a UAV is constrained by its set of actions and the set of actions is constrained by its hardware and control capabilities. An example of behavior and constituent actions is a send-report behavior, which requires a UAV to send a message to its ground station. This message could be to report compliance to commands, or to report reconnaissance data or to report health and threat status of UAV. The sequence of actions involved would be create-message, send-message and receive-acknowledgement. This information is stored in a control data packet and passed onto the output interface.

Output interface: The output interface module provides the connection between the intelligent part of the real time supervisor – the perceptor and response modules – and platform-specific subsystems such as flight control, communication systems, and payload instruments. The output interface receives control inputs from the Response controller and provides that information to the lower level controllers (Figure 3) in the form of set points.

The Real Time Supervisor architecture is scalable so that multiple instances can be arranged hierachically to distribute processing of internal self-health, external threats, and mission planning. Figure 5 shows this cascade of hierarchical controllers.

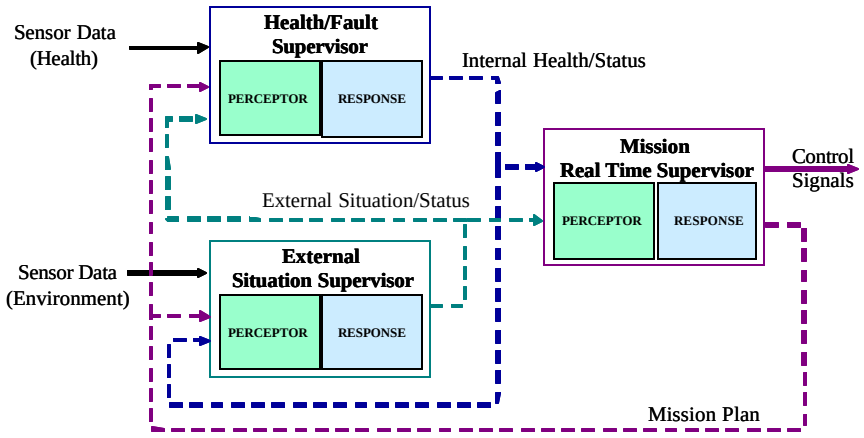

Figure 5: Multiple instances of the Real Time Supervisor can be arranged in a hierarchical structure to distribute fault detection and mission planning.

4. Real Time Supervisor: Implementation

The implementation of this architecture into a real time supervisor for fault detection, requires:

1) Hardware (UAV) Selection – Select a UAV(s), which can support the mission envisioned in the introduction followed by identification of the UAV's sensor hardware, communication system, actuators, flight capabilities and its continuous time low level controllers.
2) Identification of critical UAV systems – Identify UAV systems (propulsion, communication, surveillance etc) that need to be monitored during operation for faults and for mission performance.
3) Creation of a finite set of perception outputs (stimuli) – The perception module will create stimuli and feed it to the response module. The stimuli are in effect the perception modules assessment of the UAV's health and fault status, the UAV's environment and mission status. The nature of the stimuli is dependent on the available sensors and systems. For example it is difficult to detect fire autonomously, without heat sensing or video imaging. In the absence any such sensor a fire stimulus cannot be created by perception.
4) Creation of a finite set of behaviors – Behaviors are created based on the mission needs and stimuli from perception. However a behavior can be created only if appropriate UAV actions are available to execute the particular behavior.
5) Identification of UAV actions – Actions that the UAV can perform by virtue of its hardware capabilities, e.g. send message, go to a specific point.

4. 1 UAV Selection

Table 2 shows a short list of UAVs surveyed to identify a suitable UAV platform that will meet the mission goals. The UAVs shown meet the following criteria:

- DOD Technology Readiness level 9,
- Priced below $100,000 and
- Weighing less than 100 lbs

These conditions are consistent with the mission goals, for this project. The exception is the Schiebel Camcopter which is included because of its exceptional capability, despite being priced well above $100,000. The price for BAI Aerosystems' Javelin varies depending on the types of sensor payload, but meets the price limit. The Advanced Ceramics prototype fixed wing UAV was not included, owing to lack of enough information.

SMI is currently in contact with BAI Aerosystems' for a potential tie up to further explore embedding control and fault detection capabilities in small UAVs. At this stage of the project, the UAVs selected as application platforms are the Yamaha Rmax and the BAI Aerosystems' Javelin. While the two have very different flight dynamics, they have a high level of similarity in terms of mission capability and best meet the requirements of the project mission. In this report the Rmax is chosen to develop the methodology for implementation of the behavior based architecture. Rmax is chosen over BAI Aerosystems' Javelin because it is commercially available. It can be substituted for by Javelin without too much modification, if SMI develops a tie up with BAI Aerosystems. Javelin has the added advantage of coming with a C-programmable control, which makes it Matlab™ compatible.

As an example for demonstration, consider the critical subsystems for one particular UAV – the Yamaha Rmax. The Rmax has following critical systems:

1. Power System – For Rmax the power source is 246 cc, 21 HP two stroke engine with an attached rotor and Javelin has a battery driven electrical motor that drives the propellers.
2. Navigation System – Rmax navigates using main and tail rotor.
3. Communication system – A full duplex modem is used to communicate with the Ground station and other UAVs. Separate channels are used for sending and receiving messages.
4. Sensor System – The sensor system consists of GPS device, altitude sensor and IR sensors.
5. Surveillance System – The surveillance system consist of digital still camera and video camera.

Table 1: UAV comparison

• Model/Type • Manufacturer	Size (ft)	Weight (lbs)	Max Payload (lbs)	Propulsion	Service Ceiling (ft)	Endurance (Hrs)	Range (mi)	Cost ($)	Sensors	Communication	Remarks
• Aerolight / Fixed Wing • Aeronautics Israel	13 wingspan	80	20	6-11 HP piston	10,000	5	30	N/A	Pan tilt zoom optical camera		Cost not known; Overseas Manufacturer
• Rmax / Helicopter • Yamaha	12 x 2.5 x 3.5	128	66	21 HP piston, 246 cc	328	1	N/A	76K	Altitude, GPS, IR, Video and Still camera	Modem	Flexible; Suited for project application; Medium Cost;
• Javelin / Fixed Wing • BAI Aerosystems	8 x 6	15		Battery	1000	2	1-5		GPS, Camera w zoom, telemetry support, EO/IR		light, small and low cost; long term lease available; C programmable control
• Pointer / Fixed wing • Aerovironment	9 x 6	10 lbs	N/A	Battery; Turboprop	500	1	3	80k	IR, GPS, TV camera		Auto Navigation capability; Well defined behaviors like loiter, hold, waypoint
• Dragon Eye / Fixed Wing • ONR	4 x 3	5		Battery	1200	1	3	30K	GPS, EO, IR		ONR product; IOC* expected 6/03; Autopilot, waypoint guided
• Camcopter / Helicopter • Schiebel Tech.	Rotor dia – 10 Length 9	95	55	15 HP piston	10000	6	10 - 100	500K	GPS, Video, IR		Too expensive

*IOC – Initial Operational Capacity

The real time supervisor will monitor these systems for faults and fault level. In addition the supervisor will also monitors the environment for external threats like obstacles or approaching obstacles. The level of the fault determines the course of action. Some faults are catastrophic and the supervisor embedded in the UAV simply reports the fault to the ground station. For example, an engine failure will eventually lead to crash, however the ground station real time supervisor can reorganize the task to compensate for the loss of the UAV. On the other hand communication system failure does not imply a loss of UAV and the UAV will continue operating.

4.2 Fault Detection and Perception Outputs

The granularity of the fault tolerance system is determined by system complexity and redundancy. A large airplane has many systems like propulsion, navigation, flight control and climate control. Within these systems there are redundant components. A micro UAV, owing to its small size, low cost and light weight has much simpler propulsion and flight control, with almost no component level redundancy. The redundancy in a team of UAVs is at the UAV level. Also, individual actuator faults i.e. physical degradations are difficult to detect due to lack of sensors. Therefore, fault detection system has a much lesser granularity, compared to an aircraft. In this project the UAV faults detected will be abstracted to system level faults that cause functional degradation. The fault detection will be at a functional level. In Phase II the fault detection will provide an assessment of the Mobility, Maneuverability and Survivability of UAVs and the system. Based on the composition of the Rmax, the UAV is assumed to have the following sensing devices:

1. GPS
2. Altitude sensor
3. Infrared (IR) sensors
4. Engine/motor tachometer
5. Digital still camera
6. Video camera
7. Communication modem

Data from the first five sensing devices is used for autonomous operation, fault detection, threat detection and control of the UAV and its critical systems listed in the previous sub-section. The still and video camera perform the surveillance function and store and/or relay information to the ground station. The communication modem is used for surveillance image transmission and communicating with ground system and other UAVs. For the purpose of the Phase I simulation and demonstration, the perceptor module is designed to detect a limited number of faults as might be present in a UAV platform such as the Yamaha Rmax. Table 1 describes the faults considered in the current effort, whether the fault is internal or external in nature, and how information about the fault it inferred.

The perceptor module identifies one of the fault conditions listed in Table 1 from the available sensor or control system inputs, it notifies the response module of the existence of that condition. The response module then recommends corrective or evasive action to the lower level controllers, or recommends a change in the current mission to insure that the fault state is not encountered

Table 2: Perceptor fault states are either internal or externally stimulated and are inferred from sensor measurements.

Fault	**External/ Internal**	**Indication**
Battery low	Internal	Inferred directly from flight duration
Fuel low	Internal	Inferred directly from flight duration
Propeller stall	Internal	Inferred directly from Tachometer output
Fast drop of UAV	Internal	Inferred by perception logic from GPS and altitude sensor measurement fusion
Communication blocked by obstacle	Internal	Inferred by perception logic and communication information
Communication system failure	Internal	Inferred by perception logic and communication information
Flight Control System failure	Internal	Inferred by perception logic from GPS and altitude sensor data fusion
IR/Altitude sensor fault	Internal	Inferred by perception logic through temporal sensor data
Obstacle too close	External	Inferred from IR sensor output
Obstacle approaching too fast	External	Inferred by perception logic from sensor output
Ground too close	External	Inferred from IR sensor output
Ground approaching too fast	External	Inferred by perception logic from IR sensor output, GPS and altitude

.

4.3 UAV Behaviors and Action

The behavior of the autonomous system is bounded by the responses defined by the designed in the Real Time Supervisor's response module. The ranges of behaviors or actions that can be initiated by the response module are in turn limited by the functional capabilities of the underlying subsystems. Within the context of this Phase I effort, each UAV is assumed to be able to perform the behaviors listed in Table 2.

Table 3: UAV high-level behaviors

Behavior	**Effect**
Communicate with Tactical Coordinator (TC)	Pass message to or ping a team member or acknowledge receiving message
Loiter	Maintain present coordinates
Restore Communication	Rise to a point till its above all communication obstacles and communicate with TC
Reconnaissance	Cover a particular rectangular or circular region, while collecting pictures/terrain data. The region is broken into discrete paths;
Avoid Collision	Get away from obstacle in path
Return	Return to Ground Station
Go To	Go to a particular point.

Behavior	Effect
Resume	Resume of an action, which may have been interrupted by a change in health or environment status
Send ping	Send a ping
Send report	Send a status report to TC
Send Acknowledgement	Acknowledge a command
Transit	Reach a particular point
Hover	Stay at one point
Prepare report	Prepare a communication to transmit. The communication has four parts [Time, Sender-ID, Health Status Code, Environment Status Code];
Rise	Increase its altitude until it can restore communication or avoid an object in vicinity

The high-level behaviors listed in Table 2 are too abstract for direct interaction between the response module and the lower-level control systems. Each high-level behavior can be constructed from low-level actions and can be executed in different ways using a combination of lower-level behaviors or actions. For example 'Communicate with TC' can be constructed from the following sequence of lower-level actions:

- Send-ping – Ping its team member and the leader.
- Send-report – Send status report to leader. This behavior will prepare a report and transmit it. Status report can be sent in response to a report demand or upon behavior/mission completion or when a fault or threat is detected
- Acknowledge-message-received (order/ping) – Acknowledge receiving message or ping

The execution of these low-level actions can be mapped to control inputs for individual subsystems. The list of behaviors in Table 2 is not complete and new behaviors can be added as missions become more complex. Table 3 shows lower-level actions that can be used to construct the high-level behaviors. Table 4 provides a proposed list of higher-level behaviors and corresponding sequence of actions that will execute these behaviors.

Table 4: Low-level actions for executing behaviors

Action	Execution
Resume	Resume of an action, which may have been interrupted by a change in health or environment status
Send ping	Send a ping
Send report	Send a status report to TC
Send Acknowledgement	Acknowledge a command
Transit	Reach a particular point
Hover	Stay at one point
Prepare report	Prepare a communication to transmit. The communication has four parts [Time, Sender-ID, Health Status Code, Environment Status Code];
Rise	Increase its altitude until it can restore communication or avoid an object in vicinity

Table 5: High-level behaviors and corresponding action sequences

Behavior (mode)	**Action Sequence**
Communicate with TC (Ping)	Send Ping (TC)
Communicate with TC (Report)	Prepare report, Send Report (TC)
Communicate with TC (Acknowledgement)	Send Acknowledgement (TC)
Communicate with team member (Ping)	Send Ping (team member)
Go To	Transit (coordinates)
Loiter	Transit (loiter coordinate), Hover
Restore Communication	Rise, Send Ping(TC), Send Ping(team member)[Repeat] Resume
Return to Ground Station	Transit (Ground coordinate)
Avoid Collision	Rise, Resume
Reconnaissance	Transit (ptA), transit(ptB)……, transit(Ground coordinate)

5. Simulation and Demonstration Scenario

In phase 1 a Matlab/Simulink simulation was started to test the feasibility of the architecture to incorporate logical and mathematical tools in order to detect faults and initiate reconfiguration, in a UAV. The phase I simulation runs a basic form of the Real Time Supervisor for a drone with the components shown in Figure 3. The Real Time Supervisor runs in the simulation as it should on an actual hardware application. However all its inputs from and outputs to the hardware are simulated. The inputs from the hardware are the sensor output signals. These are simulated, as is the communication with the Tactical Coordinator (TC) and the TC itself. The outputs are the set points to autopilot and other lower level controllers. The simulation does not simulate the dynamics of the drone or the environment. It assumes them through the sensor inputs.

The simulation is intended to test the suitability of the architecture and algorithms to:

1) function in a hierarchical distributed system
2) recognize functional level faults
3) adapt to faults or reconfigure missions

The final goal of the simulation is to run a team of two drones and one Tactical Coordinator on a surveillance mission. This mission will be tested on micro UAVs in Phase II. During the course of the mission all the faults and threats listed in Section 4.2 will be simulated in different drones and the system will be tested for its ability to detect these faults and threats and if required change its behavior to accommodate them. In particular, during simulation the drones and the TC will use the Real Time Supervisor to meet the following requirements:

1. Drone shall recognize low fuel and battery levels and return to ground station.
2. Drone shall recognize and report sudden loss of altitude, steering system failure, propeller stall and report them to TC. This will be an indicator to the TC of an impending loss of drone. In case this is followed by loss of communication, a total loss is assumed and the TC shall reassign responsibility to the other two drones.
3. Drone shall recognize IR and altitude sensor fault and ignore its reading, or in case of redundancy, use other source of inputs.
4. Drone shall differentiate between communication failure and obstacle impaired communication. In case of obstacle impaired communication, it shall rise till it is above all obstacles and resend the communication. A total communication loss will imply continued data collection and recording, before returning to ground station.
5. Drone shall maintain a pre-specified minimal distance from ground and recognize and avoid obstacles.

The end of Phase I option simulation goal is to simulate a two UAV surveillance mission, where each UAV is assigned the task of scouting a sub part of one area. During the mission one UAV develops a non-catastrophic fault, like fuel low. The fault is successfully detected and communicated to the TC and the other UAV and the task is reconfigured to allot most of the surveillance area to the healthy UAV, thus reducing load on the faulty UAV.

The simulation is a work in progress and is intended as proof of concept. Figure 6 shows a schematic diagram of the simulation. The two drones are identical and the gray lines show communication channels. A communication simulation is required in order to simulate message passing between the four agents. All other dynamics and environmental affects are simulated through manually varying simulated sensor inputs.

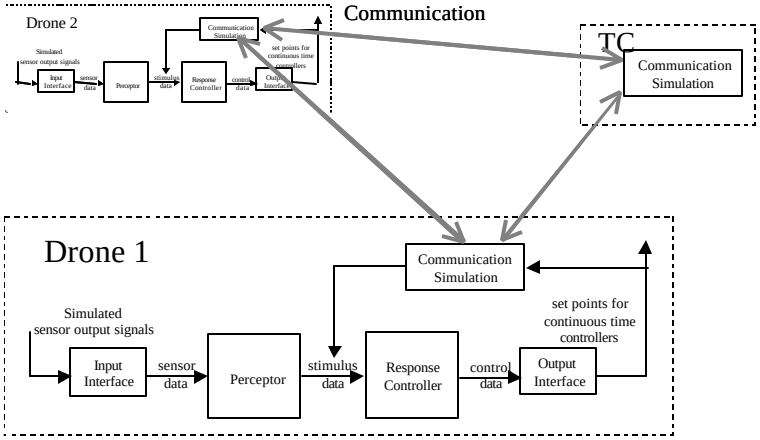

Figure 6: Simulation Schematic

5.1 Simulation Description

At present the simulink simulation includes (Figure 7):

- Simplified simulation of ground station (Tactical Coordinator), with communication and order dispatching capability;
- Simplified simulation of UAV sensor, communication to prepare interface for Real Time Supervisor;
- Fault and threat detection algorithms in perception module of Real Time Supervisor.

The simulation is a continuous time Matlab/simulink numerical simulation. It does not have three dimensional graphics or GUI, with pictures of UAVs.

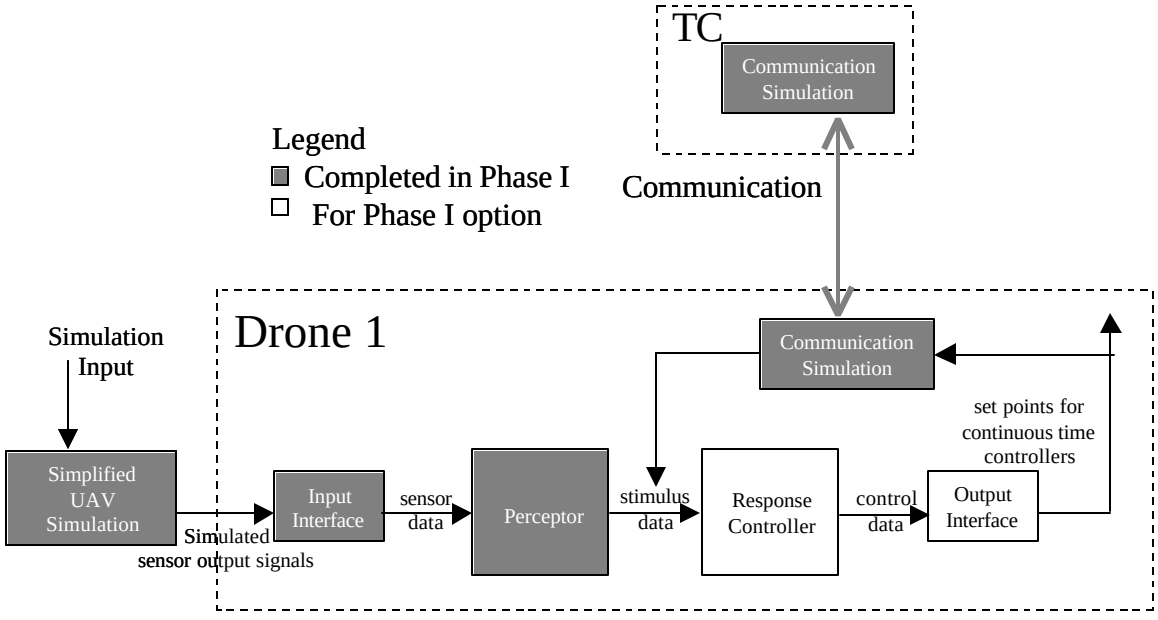

Figure 7: Simulation Status

5.1.1 Simulation Input

The Simulation Inputs are provided to the 'Simplified UAV Simulation' block. These inputs can be varied manually or in a predetermined continuous fashion, e.g., step, ramp, sinusoid, pulse etc. The simulation inputs are:

1. UAV speed
2. UAV zenith angle
3. UAV azimuth angle
4. Flight time
5. Height
6. Front obstacle distance
7. Engine RPM

5.1.2 Simplified UAV Simulation Block

The simulation inputs enter the Simplified UAV simulation block (Figure 7), which uses these inputs and initial conditions (coordinates and time) and incorporated algorithms to generate the listed (Figure 8 and next sub-section) time series outputs in the form of Simulated Sensor Output Signals, which are inputs to the Drones' Real Time Supervisor.

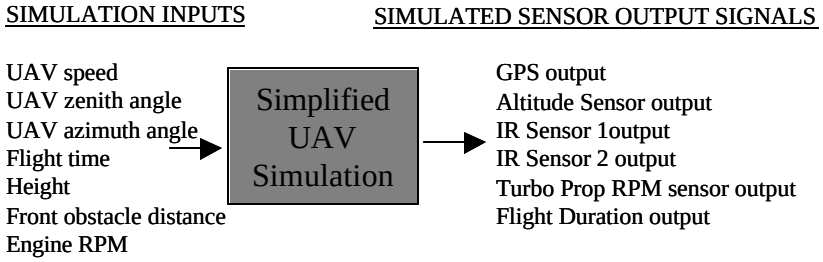

Figure 8: Simplified UAV Simulation

5.1.3 Simulated Sensor Output Signals

The inputs to each simulated real time supervisor for the Drone 1in Figure 7 are simulated sensor output signals, coming out of the simplified UAV simulation. The simplified UAV simulation provides the following continuous time sensor signals:

1. GPS output – Geographic Latitude, Geographic Longitude, Mean Altitude Above Sea Level
2. Altitude Sensor – Altitude above sea level
3. Infra Red (IR) Sensor 1- Height from ground if ground near
4. IR Sensor 2 – Distance form front obstacle if obstacle near
5. Turbo Prop RPM – Speed of main drive shaft propeller
6. Flight Time – Flight duration

5.1.4 Input Interface Block

The input interface in this application performs the function of interfacing with the sensor hardware. Its goal is to collect sensor outputs from various sensors and provide it to the

Perception block in a compatible format. In this simulation, since sensor hardware is not involved the Input Interface is a conduit between simulated sensor output and sensor data.

5.1.5 Sensor Data

The Sensor Data is output to the Perception block from the Input Interface block. The information content of Sensor Data is the same as the Simulated Sensor Output Signals

5.1.6 Perception Block

The sensor data obtained for the Input Interface is preprocessed and analyzed mathematically and logically to determine function al faults and threats.

Preprocessor

The preprocess block performs calculations and outputs the following:

1. Speed, Zenith and Azimuth Calculation – Calculated from time derivative of GPS output.
2. Descent Rate – Calculated from time derivative of altitude, IR sensor 2 and GPS output.
3. Obstacle approach rate – Calculated as IR sensor 1derivative.

Fault and Threat Analysis

The fault and threat analysis block uses algorithms and logic to identify functional level faults and threats in UAV. The inputs to this block consist of both Sensor Data and Preprocessor output. The faults identified are listed in Table 1.

Battery low and Fuel low are inferred from the flight time elapsed, using binary logic. Most batteries and fuel cartridges have a self diagnostic unit. If available that will be used to directly obtain this fault signal during actual UAV testing.

Propeller stall is inferred from Turbo RPM output.

External faults/threats are inferred as stated in Table 6.

Table 6: External Faults/Threats

External Fault/Threat	Input Source – Input Data	Logic
Obstacle too close	Sensor Data - IR Sensor 1	Temporal Binary – If-then logic over a short time series
Obstacle approaching too fast	Preprocessor - Obstacle Approach Rate	Temporal Binary
Ground too close	Sensor Data - IR Sensor 2	Temporal Binary
Ground approaching too fast	Preprocessor - Descent Rate	Temporal Binary

Before the Input Data is tested via temporal-binary logic, the IR sensor data is passed through a **variance based estimator-predictor algorithm**. This algorithm

1. Predicts the next sensor observation based on previous readings.
2. Normalizes through trending a finite number of previous sensor readings
3. Calculates the variances of this normalized data
4. If the variance is above a predetermined limit, it uses the prediction of Step 1., else it uses the senor measurement.
5. If the variance is consistently high the sensor is invalidated.

This algorithm works to remove noise and validate sensor.

The **temporal-binary** logic used for external fault detection algorithm ensures that a fault is separated from a fault like symptom. For example, there is a possibility of an obstacle crossing

the path of the UAV, as opposed to staying in the UAV path. The temporal-binary logic examines a time series of sensor reading over the past second and if they are consistently below a predetermined limit, it triggers 'obstacle too close' fault. However if the reading goes below 'briefly' and then recovers to safe limit, indicating a passing object, no fault is triggered.

The variance based estimator predictor is also used in the Fault Detection Tree shown in Figure 9. This tree also uses a moving window fast Fourier transform to detect very low frequency variation. A low frequency variation in the sensor readings corresponds to inertial dynamics and implies that the UAV is flying in a consistently haphazard fashion. This in turn implies loss of flight control. The tree in Figure 9 is used to detect: ***Fast drop of UAV, Flight Control System failure, IR/Altitude sensor fault***.

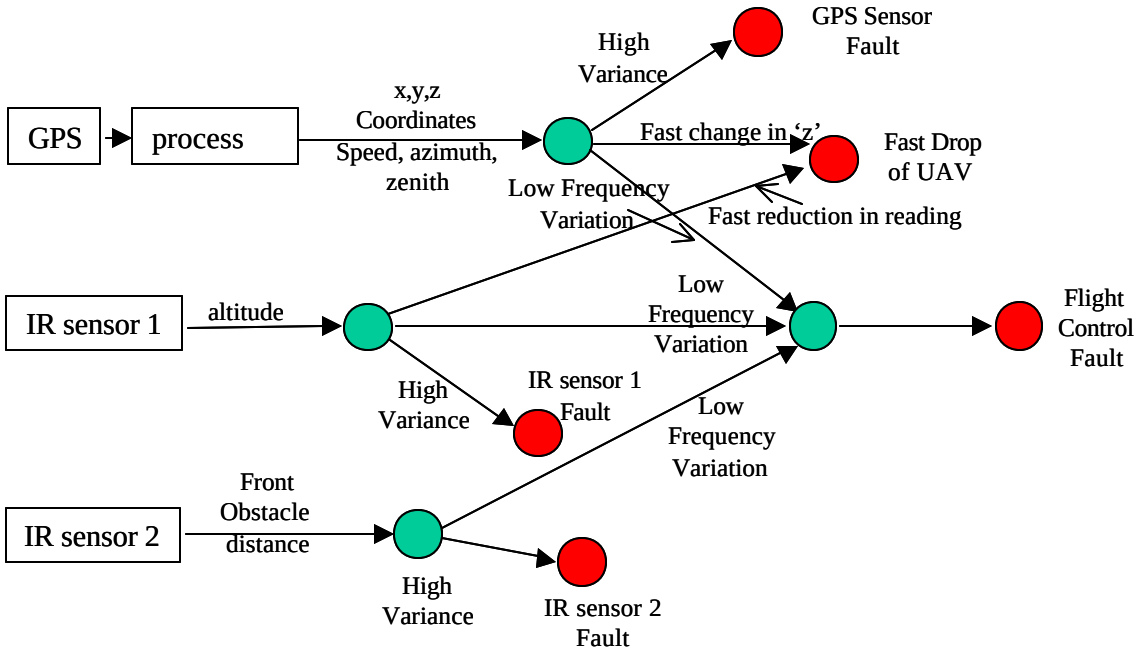

Figure 9: Fault Detection Tree

The logic for blocked communication and communication failure will be implemented in Phase I option.

5.1.7 Communication Block

Communication is simulated as a Discrete Event system. The communication simulation is identical in both the Tactical Coordinator and the UAV drones. The communication is assumed as having two separate channels, one for sending and one for receiving. Messages sent from UAVs have a sender ID, time, functional fault status, threat status and message acknowledgement. Each functional fault has an associated number and threat conditions have an associated number. Message acknowledgement can be an acknowledgement for a received message or a request for an acknowledgment from receiver or empty. Communications from the Leader/TC have commands, instead of fault status and threat status.

The communication process is event driven. The receive channel is idle till an 'incoming' message event sends it to receive state. If the message requires an acknowledgement, the reply state becomes active, which triggers an event to the sending channel, which sends an acknowledgement of receiving message. The sending may also be triggered by a fault event.

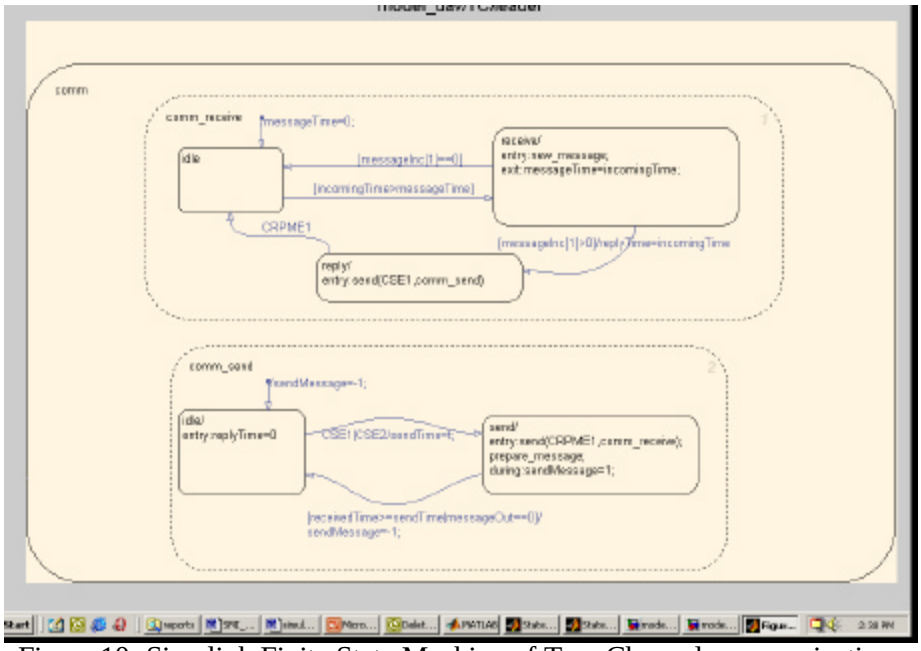

Figure 10: Simulink Finite State Machine of Two Channel communication

The simulation is a work in progress and it will be completed during Phase I option. The Response Controller block will be added to the UAV simulation and it will be tested for fault detection and reconfiguration. Initial testing of the Fault algorithms and Fault logic were encouraging. More complete tests will be performed and reported during Phase I option.

6. Conclusions and Future Work

During Phase I, SMI and ARL accomplished the following:

1. Problem Definition and Refinement – Identified cooperative, hierarchical micro UAV team missions as a target area for fault detection and reconfiguration. We defined a two UAV surveillance mission as the benchmark problem.
2. Architecture Development – Developed a flexible, distributed, hierarchical, intelligent control architecture for the Real Time Supervisor that will perform the function of fault detection and system reconfiguration.
3. Micro UAV survey – Conducted a survey of various available micro UAVs to identify a suitable UAV for Phase I and Phase II research work. We identified the Yamaha Rmax and BAI Aerosystems Javelin as suitable candidate UAVs.
4. Architecture Implementation for Platform UAV- Used the UAV survey and architecture development outcomes to transition the architecture to the Rmax UAV. We developed functional level fault detection algorithms, to embed in this architecture.
5. Proof of concept simulation – A Matlab/Simulink simulation was developed to demonstrate the feasibility of the architecture and algorithms. This simulation is a work in progress and it will be completed during Phase I option.

In the above five steps, steps 1, 2, form the basis of this project. Steps 3, 4 and 5 establish a blueprint for what will be done when the Real Time Supervisor is implemented on an actual team of UAVs, like the Rmax and tested against the benchmark problem, in Phase II. The Phase I

option will concentrate on development of the simulation. During Phase I option SMI and ARL will enhance the simulation by:

1. Implementing the perception block in the UAV real time supervisor.
2. Linking the supervisor and the communication block.
3. Testing both fault perception and response capability of a single UAV.
4. Expanding and testing the simulation for a two UAV and Tactical Coordinator simulation.

The long term goals for this project are discussed in the Phase II plan.

7. References

1. Albus, J.S. and Meystel, A. M., "A Reference Model Architecture for Design and Implementation of Intelligent Control in Large and Complex systems", *International Journal of Intelligent Control Systems*, Volume 1, Number 1, 1996, pp 15-30

2. Brooks, R. A., "A Robust Layered Control System for a Mobile Robot", Technical Report A.I. Memo 864, Massachusetts Institute of Technology, Boston, MA, 1992.

3. Roekel, M.W., Rivoir, R.H., Gibson, R.E., "A Behavior Based Controller Architecture and the Transition to an Industry Application," *Proceedings of the 1999 IEEE International Symposium on Intelligent Control*, pages 320-325.

4. Stover, J.A., Hall, D.L., and Gibson, R.E., "A Fuzzy-logic Architecture for Autonomous Multisensor Data Fusion," *IEEE Transactions on Industrial Electronics*, Vol. 43, No. 3, 1996.